SWSS LOGIN MODULE

TECHNICAL REQUIREMENTS

Introduction

This document is the technical response to the FAJ Login module. It will describe how the development team will implement the changes and additions to SWSS Childrens to effect the requirements. It is intended to explain to Policy Staff, who authored the requirements, how SWSS will behave once the solution is implemented, and is intended to explain to Project Staff how they might test the solution to verify its correctness.

This document is also to be used as a tool by the development team when coding the solution or maintaining it in the future. Thus this document is likely to be updated during the lifecycle of the SWSS project. Versions of this document will be maintained in PVCS, and the reader should be aware that multiple printed versions may exist.

Module Description

The SWSS application requires a login section in order to identify the user's application identity, security level and a list of programs to which the user has update privileges.

Requirements

Process Description

There is not a comparable process in the current programs of foster care, juvenile justice and adoption. Currently when staff need to find case information, they utilize the paper case file. There are not forms involved with the current process.

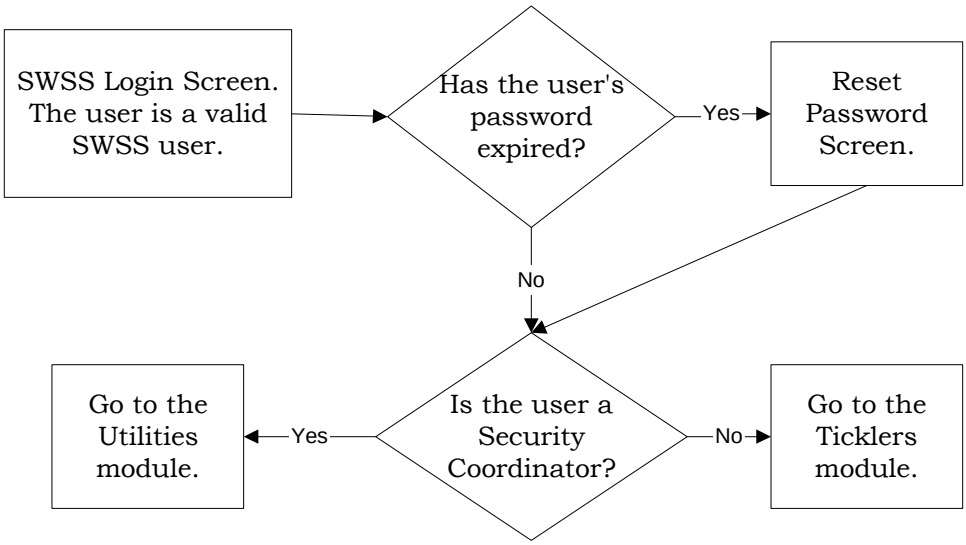
Functional Requirements

NA

Business Events

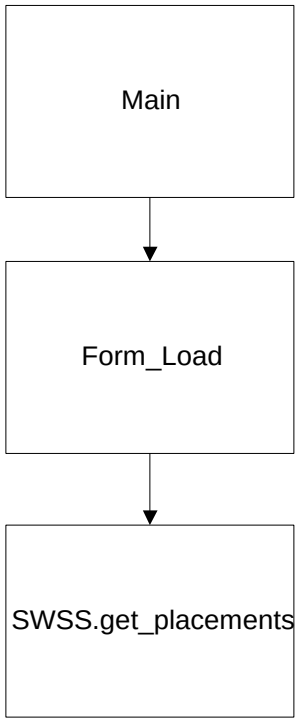
The user interacts with either initial login screen and/or the change password screen. This process is the gateway to the r4est of the FAJ application.

System Flow



List of Program Units

This would be stuff like common VB code called, the number of VB .BAS modules and form modules in the current application. Also list the Stored Procedures called. Show a "Structure Diagram" of which VB procedure calls a stored procedure or another VB procedure. Also show which stored procedures call other stored procedures.



Visual Basic Code Modules

Forms

frmAboutCFC	(Aboutcfc.frm)
frmError	(Error.frm)
frmPWD_Reset	(Pwd_Reset.frm)
frmSWSS_Login	(SWSS_Login.frm)

Code Modules

millerm7	(millerm7.bas)
swss	(swss.bas)
Login	(SWSS_Login.bas)

SQL Stored Procedures and Packages

{SWSS.SWSS_SET_DEF_SCHEMA_PKGS} (D/B team procedure)
- {swss.swss_user_exists} (D/B team function)
- {swss.swss_user_role_exists} (D/B team function)
- {swss.swss_valids} (D/B team package)
  - {swss.swss_put_line80} (D/B team procedure)
  - {swss.swss_chk_pkg_security} (D/B team function)
  - {swss.swss_pkg_grants} (D/B team procedure)
    - {swss.swss_dyn_exec} (D/B team procedure)
  - {swss.swss_dyn_execp1} (D/B team procedure)
  - {swss.swsss_extsec} (D/B team function)
- {swss.swss_internal_pkg} (D/B team function)
- {swss.swss_dyn_exec1} (D/B team procedure)

{SWSS.central_office } (D/B team package)
{SWSS.IDENTIFY_COORD} (D/B team package)
{SWSS.CENTRAL_SECURITY_ADMIN} (D/B team package)
{SWSS.loginpkg}
- FUNCTION get_user_prog_list
- FUNCTION get_user_county
- PROCEDURE get_login_user_info

{SWSS.MAIN_MENU}
- PROCEDURE get_case_info
- PROCEDURE get_update_worker_ids
- PROCEDURE get_scndry_wrkr_Ids
- PROCEDURE get_sup_sec_wrkr
- PROCEDURE get_co_adpt_supvsr_id
- PROCEDURE get_responsibility
- PROCEDURE get_load_securities
- (this is a list of all MAIN_MENU procedures and all are needed)

{SWSS.Correctionpkg}
- PROCEDURE get_correct
  - FUNCTION GetWorkerName
  - FUNCTION GetSynonymName

{SWSS.Tick_Actions}
- PROCEDURE put_wrkr_std_tick
- PROCEDURE upd_wrkr_std_tick
- PROCEDURE del_wrkr_std_tick

MODULE: SWSS_Login.frm				
FIRST CALL DEPTH	**SECOND CALL DEPTH**	**THIRD CALL DEPTH**	**FOURTH CALL DEPTH**	**FIFTH CALL DEPTH**
Form_Load	swss.PreviousInstance			
	fCenterForm			
Form_Unload	swss.CleanUpObjects			
PASSWORD_GotFocus	swss.select_text			
cmdCancel_Click	Form_Unload	swss.CleanUpObjects		
cmdok_Click	Form_Unload	swss.CleanUpObjects		
	Login.AttemptConnection			
	Login.SuccessfullConnection			

MODULE: SWSS_Login.bas				
FIRST CALL DEPTH	**SECOND CALL DEPTH**	**THIRD CALL DEPTH**	**FOURTH CALL DEPTH**	**FIFTH CALL DEPTH**
Main	frmSWSS_Login.Form_Load			
	swss.ExtractINI_Info			
AttemptConnection	swss.makeconnection			
	mbConnectionErrors	frmPWD_Reset.Form_Load		
		swss.DisplayADO_Errors		
	swss.tNthString			
mbConnectionErrors	frmPWD_Reset.Form_Load			
	swss.DisplayADO_Errors			
mbWorkerRecordNotFound	swss.PrivateINI_Register			
	swss.tGetAppPath			
	swss.tPrivateINI_GetString			
	swss.ClearINI_Sections			
	{SWSS.loginpkg.get_login_user_info}	{SWSS.loginpkg.get_wrkr_id_of_user}		
		{SWSS.loginpkg.get_user_prog_list}	{SWSS.IDENTIFY_COORD.CENTRAL_SEC}	
			{SWSS.IDENTIFY_COORD.LOCAL_SEC}	
		{SWSS.loginpkg.get_user_county}		
	swss.FillQueriedResultset			
	frmSWSS_Login.Form_Unload			
	millerm7.Store_SQL_Time			
mGetCentralOfficeCode	{SWSS.central_office.get_code}			
	swss.PrivateINI_Register			
	swss.tGetAppPath			
	swss.lPrivateINI_PutString			
mWhereToGo	swss.CommonShell			
SuccessfullConnection	swss.ExterminateSWSS			

MODULE: SWSS_Login.bas				
FIRST CALL DEPTH	**SECOND CALL DEPTH**	**THIRD CALL DEPTH**	**FOURTH CALL DEPTH**	**FIFTH CALL DEPTH**
	swss.PrivateINI_Register			
	swss.tGetAppPath			
	swss.lPrivateINI_PutString			
	millerm7. bSynchronizeSynonyms			
	mbWorkerRecordNotFound	swss.PrivateINI_Register		
		swss.tGetAppPath		
		swss.tPrivateINI_GetString		
		swss.ClearINI_Sections		
		{SWSS.loginpkg.get_login_user_info}	{SWSS.loginpkg. get_wrkr_id_of_user}	
			{SWSS.loginpkg. get_user_prog_list}	{SWSS. IDENTIFY_COORD.CENTRAL_SEC}
				{SWSS. IDENTIFY_COORD.LOCAL_SEC}
			{SWSS.loginpkg. get_user_county}	
		swss.FillQueriedResultset		
	mGetCentralOfficeCode	frmSWSS_Login.Form_Unload		
		millerm7.Store_SQL_Time		
		swss.ClearINI_Sections		
		{SWSS.central_office.get_code}		
		swss.PrivateINI_Register		
		swss.tGetAppPath		
		swss.lPrivateINI_PutString		
	mWhereToGo	swss.CommonShell		
	frmSWSS_Login.Form_Unload			

MODULE: Pwd_Reset.frm				
FIRST CALL DEPTH	**SECOND CALL DEPTH**	**THIRD CALL DEPTH**	**FOURTH CALL DEPTH**	**FIFTH CALL DEPTH**
cmdCancel_Click	frmPWD_Reset.Form_Unload			
cmdok_Click	fPWD_Change	millerm7.SendAppNameToDB		
		{SWSS.CENTRAL_SECURITY_ADMIN.MODIFY_SWSS_USER}	{SWSS.IDENTIFY_COORD.CENTRAL_SEC}	
		millerm7.Store_SQL_Time		
		swss.PrivateINI_Register		
		swss.tGetAppPath		
		swss.lPrivateINI_PutString		
		swss.makeconnection		
	frmPWD_Reset.Form_Unload			
	txtPWD_New_GotFocus	swss.select_text		
Form_Load	fCenterForm			
txtPWD_Confirm_GotFocus	swss.select_text			
txtPWD_Confirm_KeyPress	millerm7.iUNamePwdCharacter			
txtPWD_Confirm_LostFocus	millerm7.bPasswordMinLength			
txtPWD_New_GotFocus	swss.select_text			
txtPWD_New_KeyPress	millerm7.iUNamePwdCharacter			
txtPWD_New_LostFocus	millerm7.bPasswordMinLength			
fPWD_Change	millerm7.SendAppNameToDB			
	{SWSS.CENTRAL_SECURITY_ADMIN.MODIFY_SWSS_USER}			
	millerm7.Store_SQL_Time			

	swss.PrivateINI_Register			
	swss.tGetAppPath			
	swss.lPrivateINI_PutString			
	swss.makeconnection			

Screen Images

Print out the most "all-encompassing" version of the screens, which may have elements or panels that display or change based on user input. Point out any specific navigation quirks or on screen stuff, such as under what conditions various panels display or disappear. Discuss when the buttons say: "Continue" and when they say "Continue Flow" or "Print 5S".

The Screens for SWSS Login/StaffProfiles application.

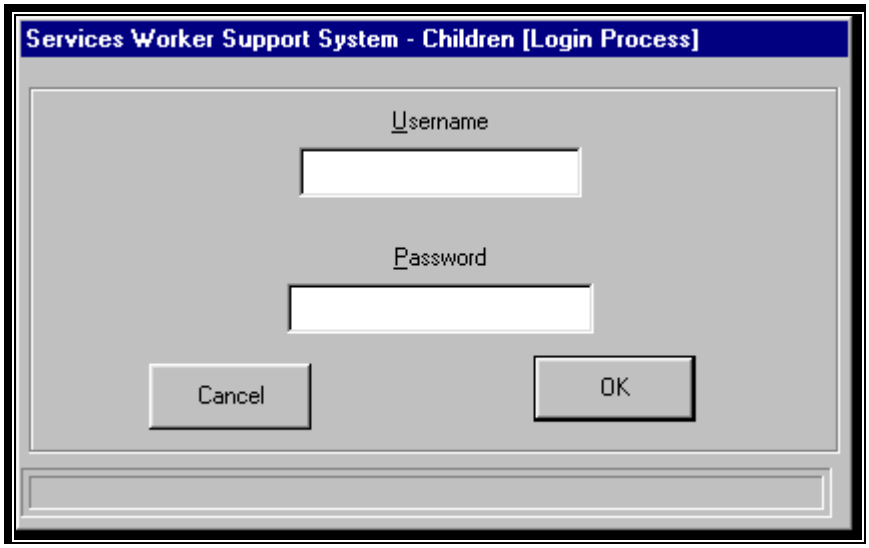


Figure 1: The Login Screen

This very simple screen lets the user type in their username and password into the appropriate fields and then either select the OK button or press the Enter key. Different things can go wrong during the login process, however.

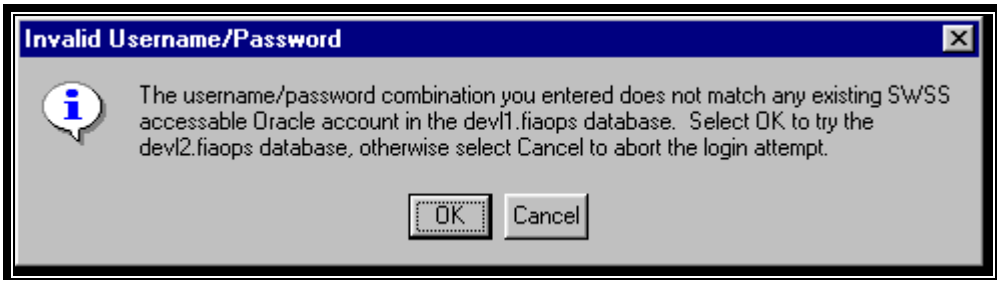


Figure 2: Invalid Username/Password message

If the user enters an invalid username/password combination, this message will come up. If the database node is unavailable, this message might also keep popping up. Selecting the OK button will retry the database connection on the next node listed in the user's SWSS_INI.INI file. Selecting the Cancel button gives the user the opportunity to re-enter their username and password (in case he/she "fat-fingered" it).

Figure 3: Expired Password Message

If any of the conditions from this message box are met, the user will see this message box. Selecting the OK button will bring up the next screen.

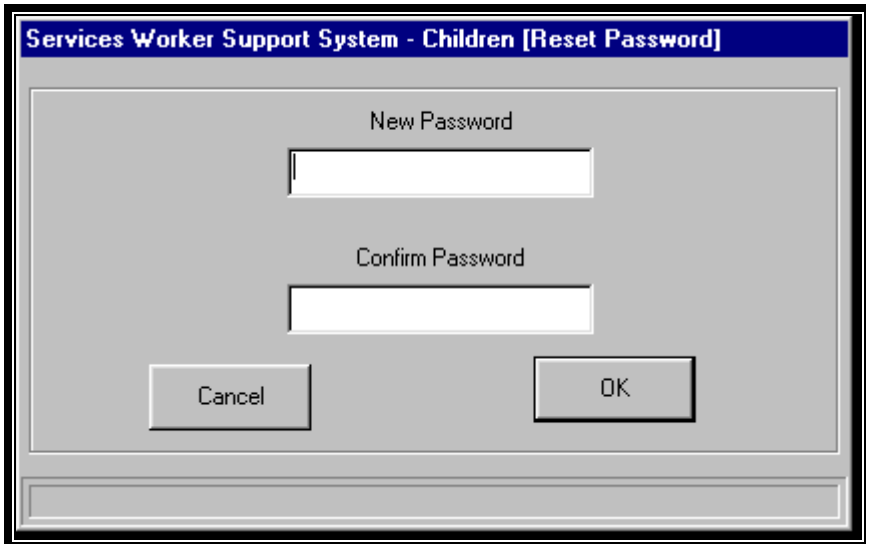


Figure 4: The Reset Password Screen

The user may then change his/her password on this screen. The password entered in the Confirm Password field must match the New Password field. However, the case of the letter combinations does not have to match. Also, the user is only permitted to use letter and numeric characters as well as the underscore character. The user can also cancel the password-reset attempt. This action, however, will prevent the user from gaining access to SWSS.

Figure 5: Locked Oracle Account Message

This message appears after a user has correctly typed his/her username but incorrectly typed the password at least three times. We also now know that this "three strikes" counter is cumulative between SWSS sessions. However, once the user successfully logs into the system, the "counter" is reset.

Report (output) Images

NA

Data Elements

For every element on a screen(s):

Form Field Description	Data Source	Data Target	Target Type	Constraints	Reqd
Login Process Screen:	*The fields on this screen do not get values from the database. They also don't change any database data.*				
Username	swss.worker.login_name				
Password	*Checks the password for the Oracle account assigned to Username*				
Reset Password Screen:					

New Password	*The values from these fields are used to change the password of the Oracle account specified in the Username field of the Login Process Screen*	Must have between 6 - 8 characters. Those character may only be the underscore character ("_"), digits 0-9 and letters A-Z (small and caps) or user may use the Backspace key - all other keystrokes are invalid	
Confirm Password		Same restrictions as the "New Password" field, plus, the value of this field must also match the value in the "New Password" field.	

Data Elements not displayed:

When calling "SWSS.SWSS_SET_DEF_SCHEMA_PKGS"…	dba_users.username dba_role_privs.grantee dba_role_privs.granted_role	
gtWorkerID	swss.worker.worker_id	Y
gtPrograms	swss.load_program.program_code swss.load.provider_mgt_ind	Y
gtUserCountyCode	swss.load.county_no	
gtCentOfficeCode	swss.central_office.G_CENTRAL_OFFICE_CODE (this is a global constant stored on the central_office package)	

Integration with Existing System

This is the first module the user's interact with in the FAJ Application. The only thing necessary at this point is a SWSS_INI.ini file with a reference to the correct database nodes.

Module Dependencies

Login is dependent upon the SWSS User profile.

Database Subject Area

- Note: The interfaces and descriptions for all the Oracle packages, procedures and functions listed, as belonging to the D/B Team, in the 'Visual Basic Module' section can be (will be) found in the technical requirement document for the Security Login project.

- PROCEDURE LOGINPKG.get_login_user_info
  (i_login_name IN security.login_name%TYPE,
  o_worker_id OUT worker.worker_id%TYPE,
  o_programs OUT VARCHAR2,
  o_county_no OUT load.county_no%TYPE,
  o_security_code OUT security.security_code%TYPE
  );
  Gather most common userdata for the SWSS_INI.ini file.

- FUNCTION LOGINPKG.get_user_county
  (i_user_worker_id worker.worker_id%TYPE)
  RETURN load.county_no%TYPE
  This function returns the highest numbered county that a user has in an active load number.

- FUNCTION LOGINPKG.get_user_prog_list
  (i_user_worker_id worker.worker_id%TYPE)
  RETURN VARCHAR2
  This function gathers all of the program codes the user has access to and returns this list in a string.

  PROCEDURE Main_Menu.get_case_info
  (i_log_id IN swss_case.log_id%TYPE,
  SWSSINIDataCursor IN OUT MAIN_MENU.SWSS_INI_Data_Cursor)
  Gather most data for the SWSS_INI.ini file from the swss_case table.

- PROCEDURE Main_Menu.get_update_worker_ids
  (i_primary_load_number IN load_case.load_no%TYPE,
  UpdateWorkerIdsCursor IN OUT MAIN_MENU.SWSS_INI_Update_Worker_Ids_Cur)

  RETRIEVE CASE INFORMATION (NOT FOUND IN THE GET_CASE_INFO
  PROCEDURE OF THE MISCPKG PACKAGE) BASED UPON A CASE'S LOG NUMBER.
  PRIMARY WORKER'S LOAD NUMBER PASSED TO PROCEDURE
  TABLE OF UPDATE CAPABLE WORKER IDs
  THIS PROCEDURE ONLY WORKS IF EACH WORKER IS ONLY ASSIGNED TO ONE
  SUPERVISOR. IF THAT IS NOT THE CASE, THIS PROCEDURE MAKE NOT PRODUCE THE CORRECT RESULTS.

- PROCEDURE Main_Menu.get_scndry_wrkr_IDs --originally named get_secondary_worker_Ids
  (i_log_id IN swss_case.log_id%TYPE,
  SecondaryIdCursor IN OUT MAIN_MENU.SWSS_INI_Secondary_Id_Cursor)
  GET LIST OF SECONDARY WORKER IDs

- PROCEDURE Main_Menu.get_sup_sec_wrkr
  (i_log_id IN swss_case.log_id%TYPE,
  o_sup_wrkr_id OUT load_worker.worker_id%TYPE)

  This is the first procedure "broken out" of the get_scndry_wrkr_IDs procedure. It returns the worker_id of the supervisor of the secondary worker assigned to the case, if that worker has a supervisor.

- PROCEDURE Main_Menu.get_co_adpt_supvsr_id
  (i_log_id IN swss_case.log_id%TYPE,
  o_sup_wrkr_id OUT load_worker.worker_id%TYPE);
  CoADPT_SupCursor IN OUT MAIN_MENU.Co_ADPT_Sup_Cursor)

  This is the second procedure "broken out" of the get_scndry_wrkr_IDs procedure. It returns the worker_id of the county adoption supervisor in the secondary worker's county.

- PROCEDURE Main_Menu.get_responsibility
  (i_log_id IN load_case.log_id%TYPE,
  i_login_worker_id IN load_worker.worker_id%TYPE,
  o_responsibility OUT load_case.responsibility%TYPE,
  o_security_level OUT security.security_code%TYPE)

  GET LOGIN USER'S RESPONSIBILITY TO SELECTED CASE.

PROCEDURE get_load_securities
  (i_login_worker_id IN load_worker.worker_id%TYPE,
  SWSSINISecurityLevelCursor IN OUT MAIN_MENU.SWSS_INI_Sec_Level_Cur)

  Gets all load numbers and their corresponding security codes for a particular worker_id

- PROCEDURE CORRECTIONPKG.get_correct
  (i_county IN swss_case.county_no%TYPE,
  i_log_id IN correction.log_id%TYPE,
  i_person_id IN correction.person_id%TYPE,
  i_table_name IN correction.swss_table%TYPE,
  i_column_name IN correction.swss_column%TYPE,
  GetCorrectionCursor IN OUT Correction_Cursor);

- FUNCTION CORRECTIONPKG.GetworkerName
  (i_login_name IN correction.login_name%TYPE,
  i_correction_date IN correction.correction_date%TYPE)
  RETURN VARCHAR2;
  get worker name at time of correction

- PROCEDURE CORRECTIONPKG.get_correct
  (i_county IN swss_case.county_no%TYPE,
  i_log_id IN correction.log_id%TYPE,
  i_person_id IN correction.person_id%TYPE,
  i_table_name IN correction.swss_table%TYPE,
  i_column_name IN correction.swss_column%TYPE,
  GetCorrectionCursor IN OUT Correction_Cursor)

- FUNCTION CORRECTIONPKG.GetSynonymName
  (i_table_name IN column_synonym.table_name%TYPE,
  i_column_name IN column_synonym.column_name%TYPE)
  RETURN VARCHAR2;

- PROCEDURE CORRECTIONPKG.Get synonym name
  (i_login_worker_id IN load_worker.worker_id%TYPE,
  SWSSINISecurityLevelCursor IN OUT MAIN_MENU.SWSS_INI_Sec_Level_Cur)

- PROCEDURE Tick_Actions.put_wrkr_std_tick
  (CountyNo IN WORKER_STD_TICKLER.COUNTY_NO%TYPE
  , WorkerID IN WORKER_STD_TICKLER.WORKER_ID%TYPE
  , TicklerID IN WORKER_STD_TICKLER.STD_TICKLER_ID%TYPE
  , LogID IN WORKER_STD_TICKLER.LOG_ID%TYPE
  , CreateDate IN WORKER_STD_TICKLER.CREATE_DATE%TYPE
  , DueDate IN WORKER_STD_TICKLER.DUE_DATE%TYPE
  , CaseNum IN WORKER_STD_TICKLER.CASE_NUM%TYPE
  , DelInd IN WORKER_STD_TICKLER.DELETE_IND%TYPE
  );

This procedure performs an insert on the TICKLER Table.

- PROCEDURE Tick_Actions.upd_wrkr_std_tick
  (CountyNo IN WORKER_STD_TICKLER.COUNTY_NO%TYPE
  , WorkerID IN WORKER_STD_TICKLER.WORKER_ID%TYPE
  , NewWorkerID IN WORKER_STD_TICKLER.WORKER_ID%TYPE
  , TicklerID IN WORKER_STD_TICKLER.STD_TICKLER_ID%TYPE
  , LogID IN WORKER_STD_TICKLER.LOG_ID%TYPE
  , CreateDate IN WORKER_STD_TICKLER.CREATE_DATE%TYPE
  , NewDueDate IN WORKER_STD_TICKLER.DUE_DATE%TYPE
  , CaseNum IN WORKER_STD_TICKLER.CASE_NUM%TYPE
  , DelInd IN WORKER_STD_TICKLER.DELETE_IND%TYPE
  );

This procedure performs an update on the TICKLER Table.

- PROCEDURE Tick_Actions.del_wrkr_std_tick
  (CountyNo IN WORKER_STD_TICKLER.COUNTY_NO%TYPE
  , WorkerID IN WORKER_STD_TICKLER.WORKER_ID%TYPE
  , TicklerID IN WORKER_STD_TICKLER.STD_TICKLER_ID%TYPE
  , LogID IN WORKER_STD_TICKLER.LOG_ID%TYPE
  , CreateDate IN WORKER_STD_TICKLER.CREATE_DATE%TYPE
  , DueDate IN WORKER_STD_TICKLER.DUE_DATE%TYPE
  , CaseNum IN WORKER_STD_TICKLER.CASE_NUM%TYPE
  , DelInd IN WORKER_STD_TICKLER.DELETE_IND%TYPE
  );

This procedure performs a delete on the TICKLER Table.

Data Warehouse

NA

Technical Issues

NA

Test Plans

No specific scenarios for testing this module was provided by Policy Staff. Testing primarily consists of determining iff the module preforms correctly given the requirements specified in section 4 of the Login User requirements document and in sections 4.4.XX, 4.5.2, 4.5.3, 4.5.5 and 4.5.6 of the Security module User Requirements.